---

title: Deploy Contracts description: Deploy your first contract on SUAVE keywords: - contract - deploy - suave - solidity

---

import RPCButton from '@site/src/components/RPCButton/index';

Deploy A Contract

:::info

SUAVE contracts can be deployed using all existing Ethereum client providers (ethers, viem, geth...) and development tool kits (forge, hardhat, brownie...). We maintain a **SUAPP examples repo** of contracts to help you get started.

:::

Deploy with Forge

Forge is a smart contract development toolchain we use in our examples. If you do not have it installed, you can get it easily by running:

bash curl -L https://foundry.paradigm.xyz | bash

To Rigil

:::info

First, get some rETH for the account you want to use from the**Rigil ETH Faucet**. Make sure you can access the private key of this account.

:::

Make a new directory and run the below to initialize it:

bash mkdir suave-contracts && cd suave-contracts/ && forge init

You can find good example contracts to learn from and test with in ourSUAPP examples repo. Choose one of these and copy-paste it into your src directory. Then run:

bash forge build

You can now deploy your compiled contract with:

bash forge create --rpc-url https://rpc.rigil.suave.flashbots.net --legacy \ --private-key <your_funded_priv_key> ./src/<your_contract_name>.sol:<your_contract_name>

You should see something like this logged to your terminal:

bash ...relevant compilation results... Deployer: 0xBE69d72ca5f88aCba033a063dF5DBe43a4148De0 Deployed to: 0xcbdF0322Cd79212e10b0dB72D775aE05B99c1796 Transaction hash: 0x9ae80af40bdafbc706108446dbbf7761a59f5bf544b46c97b9b0851dddaa3927

Locally

You can follow the exact same method as above to deploy contracts to your local SUAVE devnet. You need to have SUAVE running and use the pre-funded account we have setup for you:

bash forge create --rpc-url http://localhost:8545 --legacy \ --private-key 0x91ab9a7e53c220e6210460b65a7a3bb2ca181412a8a7b43ff336b3df1737ce12 \ ./src/<your_contract_name>.sol:<your_contract_name>

You should see something like this logged to your terminal:

bash ...relevant compilation results... Deployer: 0xBE69d72ca5f88aCba033a063dF5DBe43a4148De0 Deployed to: 0xcbdF0322Cd79212e10b0dB72D775aE05B99c1796 Transaction hash: 0x9ae80af40bdafbc706108446dbbf7761a59f5bf544b46c97b9b0851dddaa3927

SUAPP Examples

First, clone and set up the SUAPP examples repo:

bash git clone git@github.com:flashbots/suapp-examples.git

There is a convenient library called suave-std which is included as a submodule. Initialize it by running:

bash cd suapp-examples && git submodule init && git submodule update

You can compile the contracts in the repo with:

bash forge build

Locally

Now we can deploy and transact with any of the example contracts.

Once you have SUAVE running locally, cd into the relevant directory and run the main.go file. For instance, to deploy and transact with the example contract which demonstrates how updating state works for normal vs confidential requests, you can run:

bash cd examples/onchain-state && go run main.go

You should see this message printed in your terminal if successful:

bash 1. Send a confidential request that cannot modify the state 2. Send a confidential request that modifies the state

The framework.go and all of the main.go files use the Golang SDK, which you can read more about in ourresources section.

To Rigil

If you'd like to deploy one of these contracts to the Rigil Testnet instead, first ensure that you have rETH for the account you want to use from the Rigil ETH Faucet.

Then, edit the DefaultConfig of framework/framework.go to look like this (with the private key replaced):

go func DefaultConfig() *Config { return &Config{ KettleRPC: "https://rpc.rigil.suave.flashbots.net", FundedAccount: NewPrivKeyFromHex("<your_funded_priv_key_without_0x>"), } }

Now, you should be able to deploy and transact with any of the contracts in the repo. For instance, try running:

bash cd examples/onchain-state && go run main.go

You should again see output like this logged in your terminal:

bash 1. Send a confidential request that cannot modify the state 2. Send a confidential request that modifies the state

Deploy with Remix

:::warning

This method is quick, but limited. You can deploy contracts using Remix and an injected web3 provider. However, you cannot send Confidential Compute Requests to those contracts from providers like MetaMask, so it is difficult to interact with your contracts once deployed.

:::

Follow these steps to deploy a contract via Remix:

1. Add the Rigil RPC to your MetaMask or equivalent wallet and connect to it:

1. Ensure you have rETH from the Rigil ETH Faucet
2. Go to the Remix IDE and navigate to the bottom icon on the right-hand menu: "Deploy and Run Transactions".
3. Open the dropdown "Environment" menu and select "Injected Provider - MetaMask". It should showCustom (16813125) network directly below that field if this works correctly.
4. Write (or import) your contracts in the File Explorer tab, compile them, and deploy them as you usually would.